

Geometric Algebra Model of Distributed Representations

Agnieszka Patyk

Abstract Formalism based on GA is an alternative to distributed representation models developed so far — Smolensky’s tensor product, Holographic Reduced Representations (HRR) and Binary Spatter Code (BSC). Convolutions are replaced by geometric products, interpretable in terms of geometry which seems to be the most natural language for visualization of higher concepts. This paper recalls the main ideas behind the GA model and investigates recognition test results using both inner product and a clipped version of matrix representation. The influence of accidental blade equality on recognition is also studied. Finally, the efficiency of the GA model is compared to that of previously developed models.

1 Introduction

Since the early 1980s a new idea of representing knowledge has emerged by the name of distributed representation. It has been the answer to the problems of recognition, reasoning and language processing — people accomplish these everyday tasks effortlessly, often with only noisy and partial information, while computational resources required for these assignments are enormous. To this day many models have been built, in which arbitrary variable bindings, short sequences of various lengths and predicates are all usually represented as fixed-width high dimensional vectors that encode information throughout the elements. In 1990 Smolensky [14] described how tensor product algebra provides a framework for the distributed representation of recursive structures. Unfortunately, Smolensky’s tensor product does not meet all criteria of reduced representations as the size of the tensor increases with the size of the structure. Nevertheless, Smolensky and Dolan [15] have shown,

Agnieszka Patyk e-mail: patyk@mif.pg.gda.pl
Faculty of Applied Physics and Mathematics,
Gdańsk University of Technology, 80-952 Gdańsk, Poland,
Centrum Leo Apostel (CLEA), Vrije Universiteit Brussel, 1050 Brussels, Belgium.

that tensor product algebra can be used in some architectures as long as the size of a tensor is restricted. In 1994 Plate [13] worked up his Holographic Reduced Representation (HRR) that uses circular convolution and vector addition to combine vectors representing elements of a domain in hierarchical structures. Elements are represented by randomly chosen high-dimensional vectors. A vector representing a structure is of the same size as the vectors representing the elements it contains. In 1997 Pentti Kanerva [10, 11] introduced Binary Spatter Code (BSC) that is very similar to HRR and is often referred to as a form of HRR. In BSC objects are represented by binary vectors and the boolean exclusive OR is used instead of convolution. The clean-up memory is an important part of any distributed representation model as an auto-associative collection of all atomic objects and complex statements produced by that system. Given a noisy extracted vector such structure must be able to recall the most similar item stored or indicate, that no matching object had been found.

The geometric algebra (GA) model, which is the focus of this paper, is an alternative to models developed so far. It has been inspired by a well-known fact, that most people think in pictures, i.e. two- and three-dimensional shapes, not by using sequences of ones and zeroes. As far as brain functions are concerned, geometric computing has been applied thus far only in the context of primate visual system ([5], Chapters 1 and 2).

In the GA model convolutions are replaced by geometric products and superposition is performed by ordinary addition. Sentences are represented by multivectors — superpositions of blades. The concept of GA first appeared in the 19th century works of Grassmann and Clifford, but was abandoned for almost a century until Hestenes brought up the subject in [8] and [9]. The Hestenes system has recently found applications in quantum computation (Czachor *et al.* [1]–[4], [6]), which appears to be a promising leap from cognitive systems based on traditional computing.

Section 2 of this paper recalls basic operations that can be performed on blades and multivectors, using the example Kanerva [11] gave to illustrate BSC. For further details on multivectors as well as interesting exercises the reader may refer to [7, 12]. Section 3 gives rise to discussion about various ways of asking questions and investigates the percentage of correctly recognized items under two possible constructions. Section 4 introduces measures of similarity based not on only the inner product of a multivector, but also on its matrix representation. Finally, Section 5 studies the influence of accidental blade equality on recognition and Section 6 compares the performance of the GA model with HRR and BSC.

2 Geometric Algebra Model

Distributed representation models developed so far were based on long binary or real vectors. However, most people tend to think by pictures, not by sequences of numbers. Therefore geometric algebra with its ability to describe shapes is the most natural language to mimic human thought process and to represent atomic objects

as well as complex sentences. Furthermore, geometric product of two multivectors is geometrically meaningful, unlike the convolution or a binary exclusive OR operation performed on two vectors.

In this paper we consider the $\mathcal{C}\ell_n$ algebra generated by orthonormal vectors $b_i = \{0, \dots, 0, 1, 0, \dots, 0\}$ for $i \in \{1, \dots, n\}$. The inner product used throughout the paper is an extension of the inner product $\langle \cdot | \cdot \rangle$ from the Euclidean space \mathbb{R}^n . For blades $X_{<k>} = x_1 \wedge \dots \wedge x_k$ and $Y_{<l>} = y_1 \wedge \dots \wedge y_l$ the inner product $\cdot : \mathcal{C}\ell_n \times \mathcal{C}\ell_n \rightarrow \mathbb{R}$ is defined as

$$\langle X_{<k>} | Y_{<l>} \rangle = \begin{vmatrix} \langle x_1 | y_l \rangle & \langle x_1 | y_{l-1} \rangle & \dots & \langle x_1 | y_1 \rangle \\ \langle x_2 | y_l \rangle & \langle x_2 | y_{l-1} \rangle & \dots & \langle x_2 | y_1 \rangle \\ \vdots & & \ddots & \vdots \\ \langle x_k | y_l \rangle & \langle x_k | y_{l-1} \rangle & \dots & \langle x_k | y_1 \rangle \end{vmatrix} \quad \text{for } k = l, \quad (1)$$

$$\langle X_{<k>} | Y_{<l>} \rangle = 0 \quad \text{for } k \neq l \quad (2)$$

and is extended by linearity to the entire algebra.

Originally, the GA model was developed as a geometric analogue of BSC and HRR and was described by Czachor, Aerts and De Moor in [1] and [4]. Before switching from geometric product to BSC and HRR one has to realize, that geometric product is a projective representation of boolean exclusive OR. Let $x_1 \dots x_n$ and $y_1 \dots y_n$ be binary representations of two n -bit numbers x and y and let $c_x = c_{x_1 \dots x_n} = b_1^{x_1} \dots b_n^{x_n}$ and $c_y = c_{y_1 \dots y_n} = b_1^{y_1} \dots b_n^{y_n}$ be their corresponding blades, b_i^0 being equal to $\mathbf{1}$. The following examples show that geometric product of two blades c_x and c_y equals, up to a sign, $c_{x \oplus y}$

$$b_1 b_1 = c_{10 \dots 0} c_{10 \dots 0} = \mathbf{1} = c_{0 \dots 0} = c_{(10 \dots 0) \oplus (10 \dots 0)}, \quad (3)$$

$$b_1 b_{12} = c_{10 \dots 0} c_{110 \dots 0} = b_1 b_1 b_2 = b_2 = c_{010 \dots 0} = c_{(10 \dots 0) \oplus (110 \dots 0)}, \quad (4)$$

$$\begin{aligned} b_{12} b_1 &= c_{110 \dots 0} c_{10 \dots 0} = b_1 b_2 b_1 = -b_2 b_1 b_1 = -b_2 \\ &= -c_{010 \dots 0} = -c_{(110 \dots 0) \oplus (10 \dots 0)} = (-1)^D c_{(110 \dots 0) \oplus (10 \dots 0)}, \end{aligned} \quad (5)$$

the number D being calculated as follows

$$D = y_1(x_2 + \dots + x_n) + y_2(x_3 + \dots + x_n) + \dots + y_{n-1}x_n = \sum_{k < l} y_k x_l. \quad (6)$$

The original BSC is illustrated by an example taken from [11, 4] — atomic objects are represented by randomly chosen strings of bits, “ \oplus ” is a componentwise addition mod 2 and “ \boxplus ” represents a thresholded sum producing a binary vector — the threshold is set at one half of sentence chunks and a random string of bits is added in case of an even number of sentence chunks to break the tie. The encoded record is

$$PSmith = (name \oplus Pat) \boxplus (sex \oplus male) \boxplus (age \oplus 66), \quad (7)$$

and the decoding of *name* uses the involutive nature of XOR

$$\begin{aligned}
Pat' &= name \oplus PSmith \\
&= name \oplus [(name \oplus Pat) \boxplus (sex \oplus male) \boxplus (age \oplus 66)] \\
&= Pat \boxplus (name \oplus sex \oplus male) \boxplus (name \oplus age \oplus 66) \\
&= Pat \boxplus noise \rightarrow Pat.
\end{aligned} \tag{8}$$

In order to switch from BSC to HRR, the $x \mapsto c_x$ map described in [4] is used.

In the GA model, roles and fillers are represented by randomly chosen blades

$$PSmith = name * Pat + sex * male + age * 66. \tag{9}$$

The “+” is an ordinary addition and “*” written between clean-up memory items denotes the geometric product — this notation will be traditionally omitted when writing down operations performed directly on blades and multivectors. The “+” written in the superscript denotes the reversion of a blade or a multivector. The whole record now corresponds to a multivector

$$PSmith = c_{a_1 \dots a_n} c_{x_1 \dots x_n} + c_{b_1 \dots b_n} c_{y_1 \dots y_n} + c_{c_1 \dots c_n} c_{z_1 \dots z_n}, \tag{10}$$

and the decoding operation “ \sharp ” of $name$ with respect to $PSmith$ is defined as follows

$$PSmith \sharp name = name^+ * PSmith \tag{11}$$

$$\begin{aligned}
&= c_{a_1 \dots a_n}^+ [c_{a_1 \dots a_n} c_{x_1 \dots x_n} + c_{b_1 \dots b_n} c_{y_1 \dots y_n} + c_{c_1 \dots c_n} c_{z_1 \dots z_n}] \\
&= c_x \pm c_{a \oplus b \oplus y} \pm c_{a \oplus c \oplus z}
\end{aligned} \tag{12}$$

$$= Pat + noise. \tag{13}$$

It remains to employ the cleanup memory to find the element closest to Pat' — similarity is computed by the means of the inner (scalar) product. When using the decoding symbol “ \sharp ” we assume that the reader knows which model is used at the time. Therefore, there will be no variations of the “ \sharp ” symbol in BSC, HRR or in two possible GA models (depending on the way of asking a question).

For an actual example let us choose the following representation for roles and fillers of $PSmith$

$$\left. \begin{aligned} Pat &= c_{00100}, \\ male &= c_{00111}, \\ 66 &= c_{11000}, \end{aligned} \right\} \text{ fillers} \tag{14}$$

$$\left. \begin{aligned} name &= c_{00010}, \\ sex &= c_{11100}, \\ age &= c_{10001}. \end{aligned} \right\} \text{ roles} \tag{15}$$

The whole record then reads

$$\begin{aligned}
PSmith &= name * Pat + sex * male + age * 66 \\
&= c_{00010}c_{00100} + c_{11100}c_{00111} + c_{10001}c_{11000} \\
&= -c_{00110} + c_{11011} + c_{01001}.
\end{aligned} \tag{16}$$

The decoding of $PSmith$'s *name* will produce the following result

$$\begin{aligned}
name^+ * PSmith &= c_{00100} + c_{11001} - c_{01011} \\
&= Pat + noise = Pat'.
\end{aligned} \tag{17}$$

At this point, inner products between Pat' and the elements of the clean-up memory need to be compared. Item in the clean-up memory yielding the highest inner product will be the most likely candidate for Pat

$$\langle Pat | Pat' \rangle = c_{00100} \cdot (c_{00100} + c_{11001} - c_{01011}) = 1 \neq 0, \tag{18}$$

$$\langle male | Pat' \rangle = 0, \tag{19}$$

$$\langle 66 | Pat' \rangle = 0, \tag{20}$$

$$\langle name | Pat' \rangle = 0, \tag{21}$$

$$\langle sex | Pat' \rangle = 0, \tag{22}$$

$$\langle age | Pat' \rangle = 0, \tag{23}$$

$$\langle PSmith | Pat' \rangle = 0. \tag{24}$$

A question arises as to how to extract information from a multivector — should a question be asked on the left-hand-side of a multivector

$$name * PSmith, \tag{25}$$

or the right-hand-side

$$PSmith * name. \tag{26}$$

Furthermore, should we use *name* or rather $name^+$? Since we can ask about both the role and the filler, we should be able to ask both right-hand-side and left-hand-side questions according to the principles of geometric algebra. Such an approach, however, would make the rules of decomposition unclear, which is against the philosophy of distributed representations. The problem of asking reversed questions on the appropriate side of a sentence is that we should be able to distinguish roles from fillers. This implies that atomic objects should be partly hand-generated, which is not a desirable property of a distributed representation model. If we decide that a question should always be asked on one fixed side of a sentence, there is no point in reversing the blade since there is no certainty that the fixed side is the appropriate one. Independently of the hand-sidedness of questions, in test results the moduli of inner products are compared instead of their actual (possibly negative) values. For right-hand-side questions we can reformulate Equations (11) – (13) in the following way

$$PSmith \# name = PSmith * name \quad (27)$$

$$= [c_{a_1 \dots a_n} c_{x_1 \dots x_n} + c_{b_1 \dots b_n} c_{y_1 \dots y_n} + c_{c_1 \dots c_n} c_{z_1 \dots z_n}] c_{a_1 \dots a_n} \quad (28)$$

$$= \pm c_x \pm c_{b \oplus y \oplus a} \pm c_{c \oplus z \oplus a} \quad (29)$$

$$= \pm Pat + \text{noise} = Pat'. \quad (30)$$

The decoding of $PSmith$'s *name* will then take the form

$$PSmith * name = -c_{00100} - c_{11001} - c_{01011} = Pat', \quad (31)$$

resulting in $|\langle PSmith | Pat' \rangle| = |-1| = 1$. We will study the effects of asking questions in various ways in the next Section.

3 Recognition

Before we investigate the percentage of correctly recognized items, we need to introduce the following definitions. Let S and Q denote the *sentence* and the *question* respectively. Let \mathcal{A} be the set of all clean-up memory items A for which $\langle S \# Q | A \rangle \neq 0$. We will call \mathcal{A} a set of *potential answers*. Let $m = \max\{|\langle S \# Q | A \rangle| : A \in \mathcal{A}\}$ and $T = \{A \in \mathcal{A} : |\langle S \# Q | A \rangle| = m\}$. A *pseudo-answer* is an answer belonging to set T but different than the correct answer to $S \# Q$ — even if the difference is only in the meaning and not in the multivector. Of course, set T might also include the correct answer — therefore, it is called the set of (*pseudo*-)correct answers and is actually the set of answers leading to the highest modulus of the inner product. We assume that a noisy statement has been recognized correctly if its counterpart in the clean-up memory is among the (*pseudo*-)correct answers.

There are some doubts concerning how the sentences should be built — Plate [13] adds an additional vector denoting action id (usually a verb) to a sentence, e.g.

$$(\underline{eat} + eat_{agt} \circledast Mark + eat_{obj} \circledast theFish) / \sqrt{3}, \quad (32)$$

where “ \circledast ” denotes circular convolution. We will distinguish between two types of sentence constructions

- Plate construction, e.g. $eat + eat_{agt} * Mark + eat_{obj} * theFish$,
- agent-object construction, e.g. $eat_{agt} * Mark + eat_{obj} * theFish$.

The agent-object construction will often be denoted as “A-O” for short, especially in table headings. Preliminary tests conducted on the GA model were designed to investigate which type of construction suits GA better. The vocabulary set and the sentence set for these tests are included in Table 1. The sentence set is especially filled with similar sentences to test sensitivity of the GA model to confusing data. Each sentence carries a number (e.g. “(3a)”) to make further equations more compact and readable.

Table 1 Contents of the clean-up memory used in tests described throughout this paper.

Number of blades	Contents
1	A total of 42 atomic objects: 19 fillers, 7 single-feature roles and 8 double-feature roles
2	(1a) $bite_{agt} * Fido + bite_{obj} * Pat$ (2a) $flee_{agt} * Pat + flee_{obj} * Fido$
3	(3a) $see_{agt} * John + see_{obj} * (1a)$ (PSmith) $name * Pat + sex * male + age * 66$
4	(1b) $bite_{agt} * Fido + bite_{obj} * PSmith$ (2c) $flee_{agt} * PSmith + flee_{obj} * Fido$ (4a) $cause_{agt} * (1a) + cause_{obj} * (2a)$
5	(3b) $see_{agt} * John + see_{obj} * (1b)$ (5a) $see_{agt} * John + see_{obj} * (4a)$
6	(4c) $cause_{agt} * (1b) + cause_{obj} * (2a)$
7	(DogFido) $class * animal + type * dog + taste * chickenlike$ $+ name * Fido + age * 7 + sex * male + occupation * pet$
8	(1c) $bite_{agt} * DogFido + bite_{obj} * Pat$ (2b) $flee_{agt} * Pat + flee_{obj} * DogFido$ (4b) $cause_{agt} * (1b) + cause_{obj} * (2c)$
9	(3c) $see_{agt} * John + see_{obj} * (1c)$ (5b) $see_{agt} * John + see_{obj} * (4b)$
10	(1d) $bite_{agt} * DogFido + bite_{obj} * PSmith$ (2d) $flee_{agt} * PSmith + flee_{obj} * DogFido$
11	(3d) $see_{agt} * John + see_{obj} * (1d)$

(*) Each sentence carries a number (e.g. “(3a)”) to make further equations more readable.

3.1 Right-hand-side questions

In the previous Section we commented on the use of reversion and the choice of the side of a statement that a question should be asked on. The argument for right-hand-side (generally: fixed-hand-side) questions without a reversion was that rules of decomposition of a statement should be clear and unchangeable. However, the use of right-hand-side questions poses a problem best described by the following example. Let the clean-up memory contain seven roles and fillers

$$\begin{aligned}
 see_{agt} &= c_{00101}, & John &= c_{00101}, \\
 see_{obj} &= c_{01010}, & Pat &= c_{10000}, \\
 bite_{agt} &= c_{10110}, & Fido &= c_{10001}, \\
 bite_{obj} &= c_{00001},
 \end{aligned} \tag{33}$$

and two sentences mentioned in Table 1

$$(1a) \quad bite_{agt} * Fido + bite_{obj} * Pat = c_{00111} - c_{10001}, \tag{34}$$

$$(3a) \quad see_{agt} * John + see_{obj} * (1a) = -c_{00000} - c_{01101} - c_{11011}. \tag{35}$$

Let us now ask a question $(3a) \# see_{obj} = (3a) * see_{obj}$. The decoded answer

$$\begin{aligned} (3a) * see_{obj} &= (-c_{00000} - c_{01101} - c_{11011})c_{01010} \\ &= -c_{01010} + c_{00111} + c_{10001} \end{aligned} \quad (36)$$

$$= \text{noise} + bite_{agt} * Fido - bite_{obj} * Pat \quad (37)$$

results in one noisy chunks and two chunks resembling sentence (1a) but having a partially different sign than the original (1a). Furthermore,

$$\begin{aligned} (1a) \cdot ((3a) * see_{obj}) &= (c_{00111} - c_{10001}) \cdot (-c_{01010} + c_{00111} + c_{10001}) \\ &= c_{00111} \cdot c_{00111} - c_{10001} \cdot c_{10001} \end{aligned} \quad (38)$$

$$= -\mathbf{1} + \mathbf{1} = 0. \quad (39)$$

Such a situation would not have happened if we asked differently

$$see_{obj}^+ * (3a), \quad (40)$$

since $see_{obj}^+ see_{obj} = \mathbf{1}$ for normalized atomic objects.

The similarity of (1a) and $(3a) * see_{obj}$ equals zero because the nonzero similarities of blades (i.e. $\mathbf{1}$ s) belonging to these statements cancelled each other out. Cancellation could be most likely avoided, if sentence (1a) had an odd number of blades. This observation has been backed up by test results comparing the performance of Plate construction and the agent-object construction.

As expected, the agent-object construction seems to work better for sentences from which a rather simple information is to be derived, e.g. $PSmith \# name$ or $(5a) \# see_{obj}$ (Figures 1 and 2 respectively). However, when the information asked

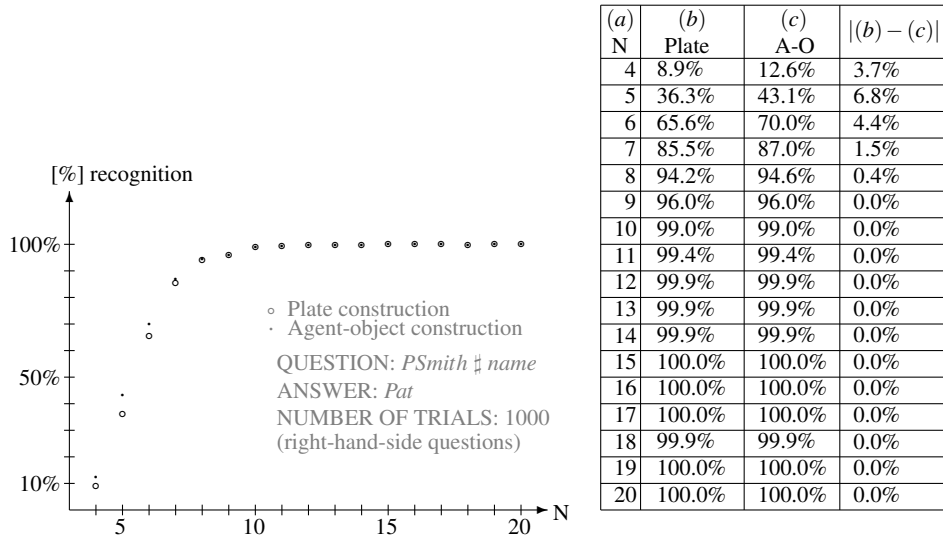
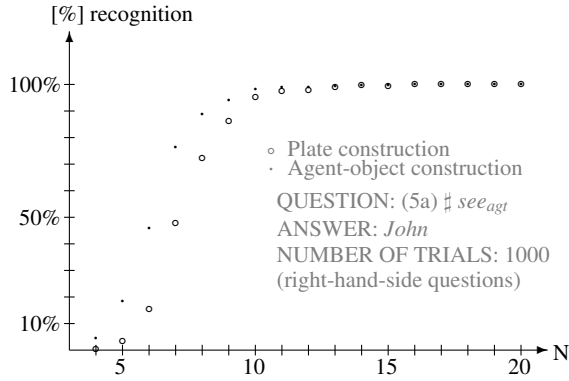
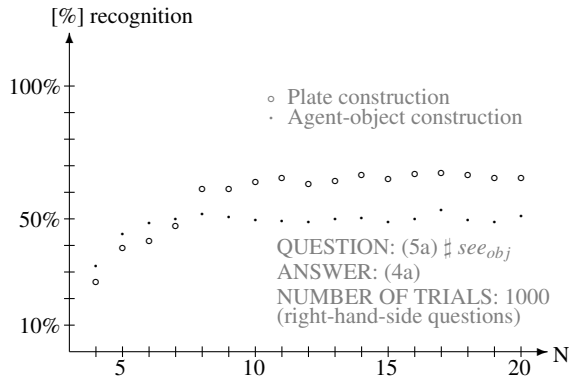


Fig. 1 Recognition test results for $PSmith \# name$.

Fig. 2 Recognition test results for (5a) $\# \text{see}_{agt}$.

(a) N	(b) Plate	(c) A-O	$ (b) - (c) $
4	0.6%	4.4%	3.8%
5	3.3%	18.5%	15.2%
6	15.3%	45.8%	30.5%
7	47.8%	76.4%	28.6%
8	72.2%	88.9%	16.7%
9	86.4%	94.3%	7.9%
10	95.1%	98.2%	3.1%
11	97.7%	98.9%	1.2%
12	98.1%	99.2%	1.1%
13	99.1%	99.6%	0.5%
14	99.9%	99.9%	0.0%
15	99.6%	99.9%	0.3%
16	100.0%	100.0%	0.0%
17	100.0%	100.0%	0.0%
18	100.0%	100.0%	0.0%
19	100.0%	100.0%	0.0%
20	100.0%	100.0%	0.0%

Fig. 3 Recognition test results for (5a) $\# \text{see}_{obj}$.

(a) N	(b) Plate	(c) A-O	$ (b) - (c) $
4	26.4%	32.2%	5.8%
5	39.2%	44.2%	5.0%
6	41.8%	48.6%	6.8%
7	47.3%	50.1%	2.8%
8	61.4%	51.7%	9.7%
9	61.2%	50.6%	10.6%
10	64.0%	49.5%	14.5%
11	65.4%	49.3%	16.1%
12	63.2%	48.7%	14.5%
13	64.3%	49.9%	14.4%
14	66.4%	50.4%	16.0%
15	64.9%	48.8%	16.1%
16	66.9%	49.8%	17.1%
17	67.3%	53.3%	14.0%
18	66.4%	49.4%	17.0%
19	65.4%	48.8%	16.6%
20	65.2%	51.1%	14.1%

was more complex, e.g. (5a) $\# \text{see}_{obj}$, the Plate construction seemed more appropriate (Figure 3). This might be so for at least two reasons:

- some of the blades belonging to the answer of (5a) $\# \text{see}_{obj}$ appear in numerous entries in the cleanup memory listed in Table 1, causing the GA model to misinterpret the answers it receives after the inner products have been computed,

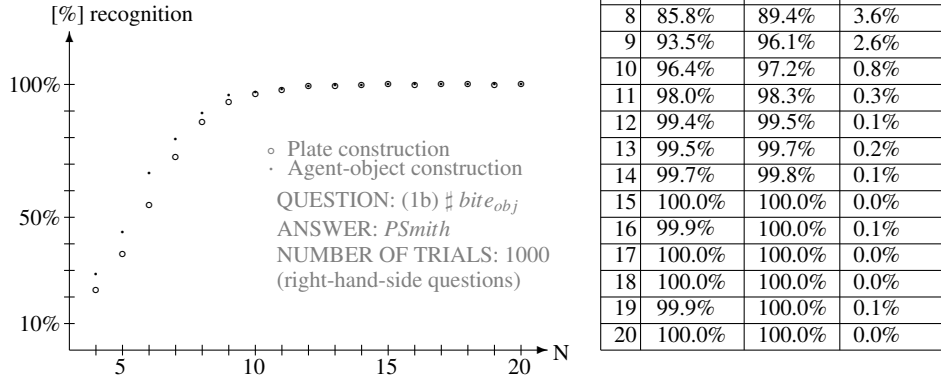


Fig. 4 Recognition test results for (1b) # *bite_obj*.

- the number of blades in (5a) # *see_obj* is uneven when using Plate construction, hence the possibility that blades' similarities cancel each other out is smaller than in case of the agent-object construction — such hypothesis would be backed up by test results depicted in Figure 4.

These hypotheses led to a conclusion that perhaps a random blade should be added to those sentences that have an even number of blades, similarly to BSC.

A correct answer might not be recognized for two reasons:

- the correct answer has an even number of blades and their similarities cancelled each other out completely because of having opposite signs - hence such an answer does not even appear within the set of potential answers,
- there are some pseudo-answers leading to a higher inner product because the similarities of blades of a correct answer cancelled each other partially.

Adding random extra blades that make the number of blades in a multivector odd (for short: *odding blades*) is a solution to the first reason why a correct answer is not recognized. Further, an *odding blade* acts as a distinct marker belonging only to one sentence (for sufficiently large data size) distinguishing it from other sentences, unlike the extra blade representing action in Plate construction which may appear in numerous sentences. Unfortunately, to address the second problem, we need to employ some other measurement of similarity than the inner product. We will show in Section 4, that Hamming and Euclidean measures perform very well in that case.

Observation of preliminary recognition test results led to a conclusion, that sentences with an even number of blades behave quite differently than sentences with an odd number of blades. In the following tests we inspected the average number

of times that blades' similarities cancelled each other out completely during the computation of similarity via inner product.

The complete cancellation of similarities takes place only when exactly half of blades of the correct answer carry a plus sign and the other half carry a minus sign. If the correct answer has $2K \geq 2$ blades, then the probability of exactly half of blades having the same sign is

$$\frac{\binom{2K}{K}}{2^{2K}} \quad (41)$$

under assumption, that the sentence set is chosen completely at random without the interference of the experimenter. Figures 5 through 7 show three examples of questions yielding an even-number blade answer and the average number of times their blades' similarities cancelled each other out completely.

3.2 Appropriate-hand-side reversed questions

Let us recall some roles and fillers

$$\begin{aligned} see_{agt} &= c_{00101}, & John &= c_{00101}, \\ see_{obj} &= c_{01010}, & Pat &= c_{10000}, \\ bite_{agt} &= c_{10110}, & Fido &= c_{10001}, \\ bite_{obj} &= c_{00001}, \end{aligned} \quad (42)$$

as well as two sentences mentioned in Table 1

$$(1a) \quad bite_{agt} * Fido + bite_{obj} * Pat = c_{00111} - c_{10001}, \quad (43)$$

$$(3a) \quad see_{agt} * John + see_{obj} * (1a) = -c_{00000} - c_{01101} - c_{11011}. \quad (44)$$

The answers to questions (3a) $\sharp see_{obj}$ and (3a) $\sharp John$ should be computed in different ways

$$(3a) \sharp see_{obj} = see_{obj}^+ * (3a) \approx (1a), \quad (45)$$

$$(3a) \sharp John = (3a) * John^+ \approx see_{agt}. \quad (46)$$

We will concentrate only on the first question

$$\begin{aligned} see_{obj}^+ * (3a) &= c_{01010}^+ (-c_{00000} - c_{01101} - c_{11011}) \\ &= c_{01010} (c_{00000} + c_{01101} + c_{11011}) \\ &= c_{01010} + c_{00111} - c_{10001} \end{aligned} \quad (47)$$

$$= \text{noise} + (1a) = (1a)'. \quad (48)$$

Only two elements of the clean-up memory are similar to $(1a)'$

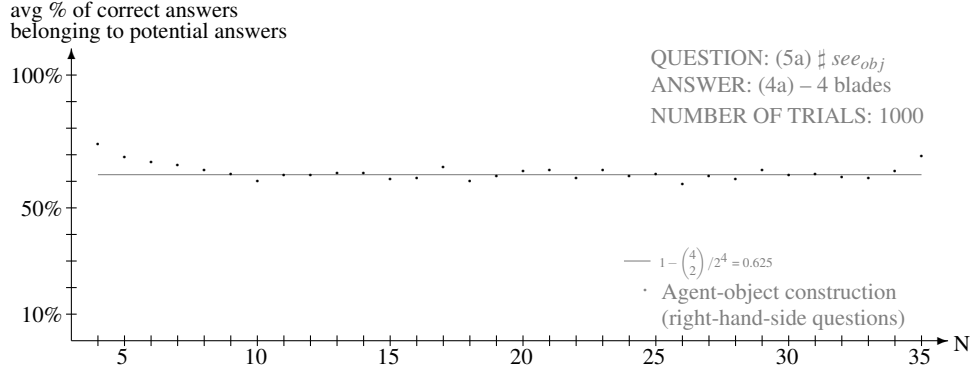


Fig. 5 The average number of times a correct answer appears within the set of potential answers ((5a) $\# see_{obj}$).

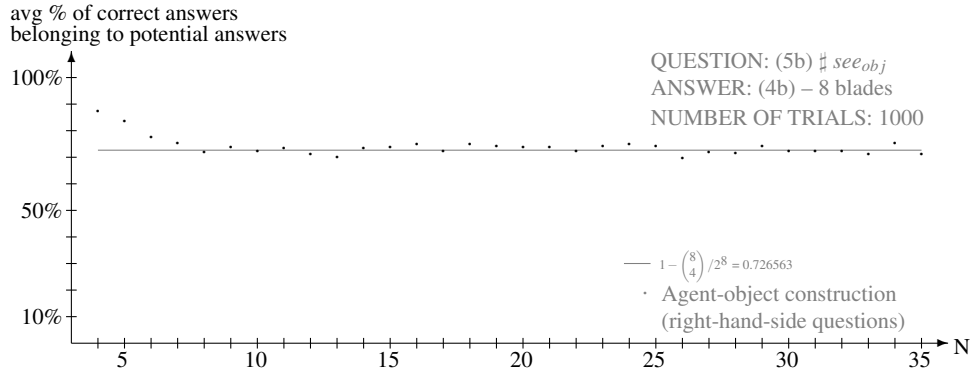


Fig. 6 The average number of times a correct answer appears within the set of potential answers ((5b) $\# see_{obj}$).

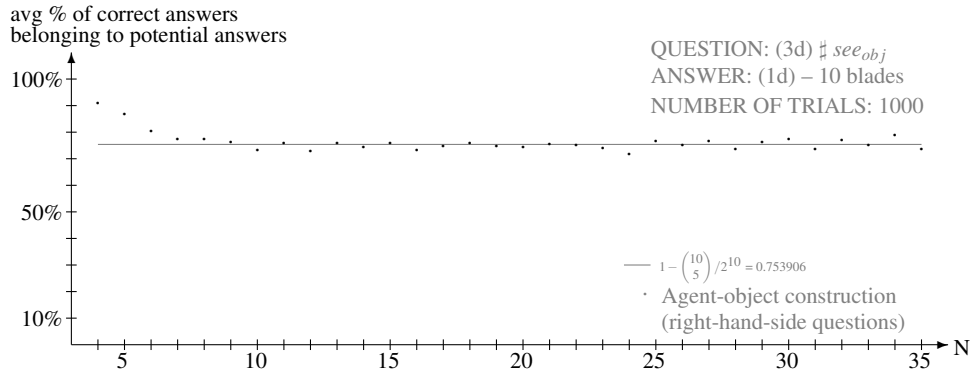


Fig. 7 The average number of times a correct answer appears within the set of potential answers ((3d) $\# see_{obj}$).

$$|\langle see_{obj} | (1a)' \rangle| = 1, \quad (49)$$

$$\begin{aligned} |\langle (1a) | (1a)' \rangle| &= |(c_{00111} - c_{10001}) \cdot (c_{01010} + c_{00111} - c_{10001})| \\ &= |c_{00111} \cdot c_{00111} + c_{10001} \cdot c_{10001}| \\ &= |-\mathbf{1} - \mathbf{1}| = 2, \end{aligned} \quad (50)$$

where see_{obj} is similar to the noise term only by accident.

Asking reversed questions on the appropriate side has one huge advantage over fixed-hand-side questions: no similarities cancel each other out neither completely nor partially while similarity is being computed, hence there is no need for adding odding vectors. For small data size blades may cancel each other out at the moment a sentence is created. Nevertheless, in all cases recognition will quickly reach 100% and the only problem that might appear is that several items of the clean-up memory might be equally similar.

4 Other measures of similarity

The inner product is not the only way to measure the similarity of concepts stored in the clean-up memory. This Section comments on the use of matrix representation and its advantages in the unavoidable presence of similarity cancellation and many equally probable answers. We will show, that comparison by Hamming and Euclidean measures gives promising results such cases.

4.1 Matrix representation

Matrix representations of GA, although not efficient, are useful for performing cross-checks of various GA constructions and algorithms. An arbitrary n -bit record can be encoded into the matrix algebra known as Cartan representation of Clifford algebras as follows

$$b_{2k} = \underbrace{\sigma_1 \otimes \cdots \otimes \sigma_1}_{n-k} \otimes \sigma_2 \otimes \underbrace{1 \otimes \cdots \otimes 1}_{k-1}, \quad (51)$$

$$b_{2k-1} = \underbrace{\sigma_1 \otimes \cdots \otimes \sigma_1}_{n-k} \otimes \sigma_3 \otimes \underbrace{1 \otimes \cdots \otimes 1}_{k-1}, \quad (52)$$

using Pauli's matrices

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (53)$$

Let us once again consider the roles and fillers of *PSmith*

$$\left. \begin{array}{l} Pat = c_{00100}, \\ male = c_{00111}, \\ 66 = c_{11000}, \end{array} \right\} \text{ fillers} \quad (54)$$

$$\left. \begin{array}{l} name = c_{00010}, \\ sex = c_{11100}, \\ age = c_{10001}, \end{array} \right\} \text{ roles} \quad (55)$$

as described in Section 2. Their explicit matrix representations are

$$Pat = c_{00100} = b_3 = \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1, \quad (56)$$

$$\begin{aligned} male = c_{00111} &= b_3 b_4 b_5 \\ &= (\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1)(\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_2 \otimes 1)(\sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1 \otimes 1) \\ &= \sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes (-i\sigma_1) \otimes 1, \end{aligned} \quad (57)$$

$$\begin{aligned} 66 = c_{11000} &= b_1 b_2 = (\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3)(\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_2) \\ &= 1 \otimes 1 \otimes 1 \otimes 1 \otimes (-i\sigma_1), \end{aligned} \quad (58)$$

$$name = c_{00010} = b_4 = \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_2 \otimes 1, \quad (59)$$

$$\begin{aligned} sex = c_{11100} &= b_1 b_2 b_3 \\ &= (\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3)(\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_2)(\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1) \\ &= \sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1 \otimes (-i\sigma_1), \end{aligned} \quad (60)$$

$$\begin{aligned} age = c_{10001} &= b_1 b_5 = (\sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_1 \otimes \sigma_3)(\sigma_1 \otimes \sigma_1 \otimes \sigma_3 \otimes 1 \otimes 1) \\ &= 1 \otimes 1 \otimes (-i\sigma_2) \otimes \sigma_1 \otimes \sigma_3. \end{aligned} \quad (61)$$

Figure 8 shows six blades making up $PSmith$ for $n = 5$ and Figure 9 shows the matrix representation of $PSmith$ for $n \in \{6, 7\}$, black dots indicate nonzero matrix entries.

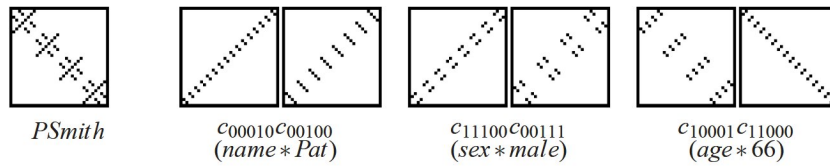


Fig. 8 $PSmith$ and its blades for $n = 5$.

The regularity of patterns placed along the diagonals is not an accident. Consider Cartan representation of blades b_{2k} and b_{2k-1} — the shortest sequence of $n - k$ σ_1 's will occur for $k = \lceil \frac{n}{2} \rceil$ (in other words, in blade b_n). Therefore each blade b_1, \dots, b_n has at least $\lfloor \frac{n}{2} \rfloor$ of σ_1 's placed at the beginning of the formula describing its representation. Hence, there are exactly $2^{\lfloor \frac{n}{2} \rfloor}$ “boxes” of patterns placed along one of the diagonals, each one of dimensions $2^{\lceil \frac{n}{2} \rceil} \times 2^{\lceil \frac{n}{2} \rceil}$. To extract a part individual for a given blade, one needs to consider only the last $\lceil \frac{n}{2} \rceil + 1$ of σ 's or unit matrices belonging to its representation — the extra σ_1 bearing the number $n - \lceil \frac{n}{2} \rceil$ is needed to preserve the direction of the diagonal — either “top left to bottom right” or “top

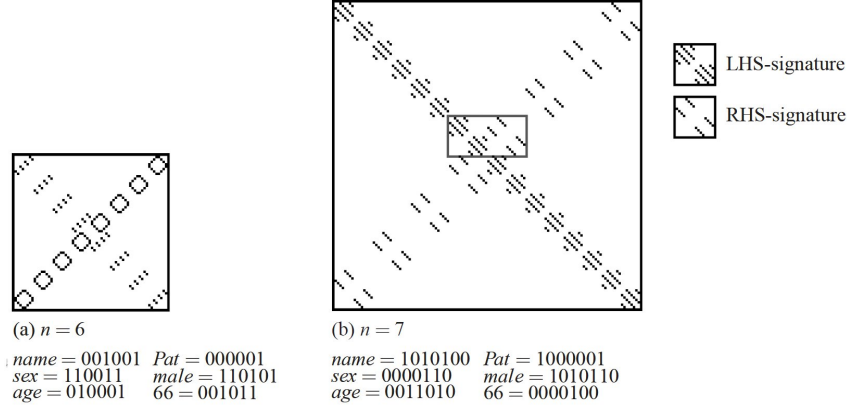


Fig. 9 An example of matrix representation of $PSmith$ for $n \in \{6, 7\}$.

right to bottom left”. If $c = b_{\alpha_1} \dots b_{\alpha_m}$ is a blade representing an atomic object in the clean-up memory, say *Pat*, then such an object has a “top left to bottom right” orientation if and only if $m \equiv 0 \pmod{2}$. Therefore we can reformulate equations (51) and (52) in the following way

$$b_{2k} = \underbrace{\sigma_1 \otimes \dots \otimes \sigma_1}_{\lceil \frac{n}{2} \rceil - k + 1} \otimes \sigma_2 \otimes \underbrace{\mathbf{1} \otimes \dots \otimes \mathbf{1}}_{k-1}, \quad (62)$$

$$b_{2k-1} = \underbrace{\sigma_1 \otimes \dots \otimes \sigma_1}_{\lceil \frac{n}{2} \rceil - k + 1} \otimes \sigma_3 \otimes \underbrace{\mathbf{1} \otimes \dots \otimes \mathbf{1}}_{k-1}. \quad (63)$$

Consider once again the representation of $PSmith$ depicted in Figure 9b. To distinguish $PSmith$ from other object we only need to store two of its “boxes” — each “box” lying along a different diagonal, Figure 9b shows such two parts. We will call the two different “boxes” *left-hand-side signatures* and *right-hand-side signatures* depending on the corner the diagonal is anchored to at the top of the matrix. It is worth noticing that signatures for $n = 2k - 1$ and $n = 2k$ are of the same size, which causes some test results diagrams to resemble step functions.

Note that the use of tensor products in GA bears no resemblance to Smolensky’s model, as the rank of a tensor does not increase with the growing complexity of a sentence.

4.2 The Hamming measure of similarity

The first most obvious method of comparing two matrices or their signatures would be to compute the number of entries they have in common and the number of entries they differ by. Let $X = [x_{ij}]$ and $Y = [y_{ij}]$ be signatures of matrices, i.e. $i \in \{1, \dots, 2^{\lceil \frac{n}{2} \rceil + 1}\}$, $j \in \{1, \dots, 2^{\lceil \frac{n}{2} \rceil}\}$. Let

$$c(x_{ij}, y_{ij}) = \begin{cases} 1 & \text{if } x_{ij} \neq 0 \text{ and } y_{ij} \neq 0, \\ 0 & \text{otherwise,} \end{cases} \quad (64)$$

$$u(x_{ij}, y_{ij}) = 1 - c(x_{ij}, y_{ij}). \quad (65)$$

Now let us count the number of *common points* and *uncommon points*

$$C(X, Y) = \sum_{i,j} c(x_{ij}, y_{ij}), \quad (66)$$

$$U(X, Y) = \sum_{i,j} u(x_{ij}, y_{ij}). \quad (67)$$

Finally, the Hamming measure for comparing the signatures of matrices computes the ratio of common and uncommon points

$$H(X, Y) = \begin{cases} \frac{C(X, Y)}{U(X, Y)} & \text{if } U(X, Y) \neq 0, \\ \infty & \text{otherwise.} \end{cases} \quad (68)$$

Such a measure of similarity is fairly fast to calculate since it does not involve computing any mathematical operations except addition and the final division of $C(X, Y)$ and $U(X, Y)$.

4.3 The Euclidean measure of similarity

The second most obvious method for computing matrix similarity is via Euclidean distance. Again, let $X = [x_{ij}]$ and $Y = [y_{ij}]$ be signatures of matrices for $i \in \{1, \dots, 2^{\lceil \frac{n}{2} \rceil + 1}\}$, $j \in \{1, \dots, 2^{\lceil \frac{n}{2} \rceil}\}$. Let

$$E(X, Y) = \begin{cases} \frac{1}{\sum_{i,j} \sqrt{|x_{ij}|^2 - |y_{ij}|^2}} & \text{if } \sum_{i,j} \sqrt{|x_{ij}|^2 - |y_{ij}|^2} \neq 0, \\ \infty & \text{otherwise.} \end{cases} \quad (69)$$

This kind of measure uses more mathematical operations requiring greater time to compute — the modulus of a complex number, multiplication and the square root. The Hamming measure involved calculating only addition and the ratio of common and uncommon points. Calculating the ratio in both measures results in those measures taking on a role of “probability” that the matrices are alike rather than describ-

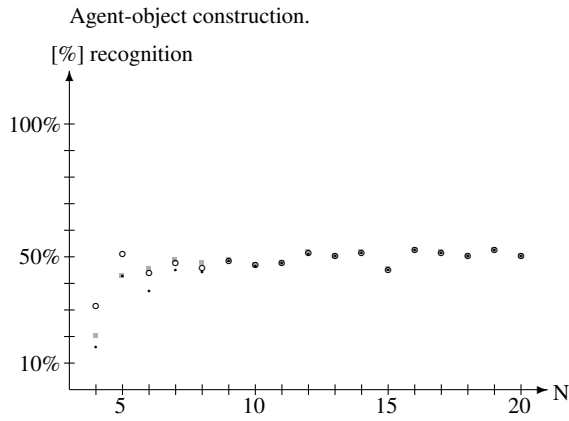
QUESTION: (4a) \sharp $cause_{obj}$

ANSWER: (2a)

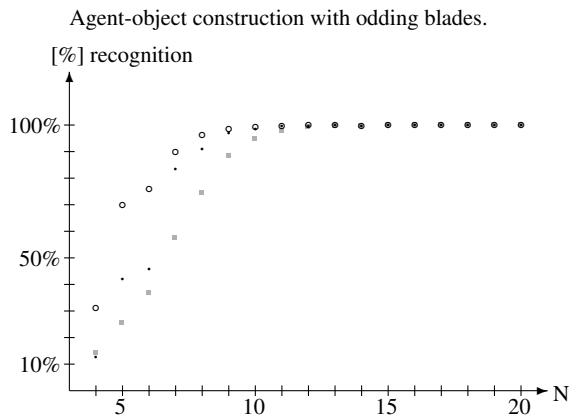
NUMBER OF TRIALS: 500

- Inner product
- Euclidean measure
- Hamming measure

(right-hand-side questions)



N	Hamming	Euclidean	Inner pr.
4	16.2%	31.4%	20.2%
5	42.8%	51.0%	42.8%
6	37.0%	43.8%	45.2%
7	45.2%	47.6%	48.6%
8	44.4%	45.8%	47.6%
9	48.4%	48.6%	48.8%
10	46.6%	47.0%	47.0%
11	47.8%	47.8%	47.6%
12	51.0%	51.6%	51.6%
13	50.4%	50.4%	50.4%
14	51.6%	51.6%	51.6%
15	45.2%	45.2%	45.2%
16	52.6%	52.6%	52.6%
17	51.6%	51.6%	51.6%
18	50.4%	50.4%	50.4%
19	52.4%	52.4%	52.4%
20	50.2%	50.2%	50.2%



N	Hamming	Euclidean	Inner pr.
4	12.6%	31.2%	14.0%
5	42.2%	70.0%	25.6%
6	45.8%	75.8%	36.8%
7	83.4%	89.8%	57.4%
8	91.0%	96.2%	74.4%
9	97.0%	98.4%	88.4%
10	98.6%	99.2%	94.8%
11	99.8%	99.8%	97.8%
12	99.6%	100.0%	99.2%
13	100.0%	100.0%	99.4%
14	99.8%	99.8%	99.4%
15	100.0%	100.0%	100.0%
16	100.0%	100.0%	100.0%
17	100.0%	100.0%	100.0%
18	100.0%	100.0%	100.0%
19	100.0%	100.0%	100.0%
20	100.0%	100.0%	100.0%

Fig. 10 Recognition test via inner product, Hamming and Euclidean measure for (4a) \sharp $cause_{obj}$.

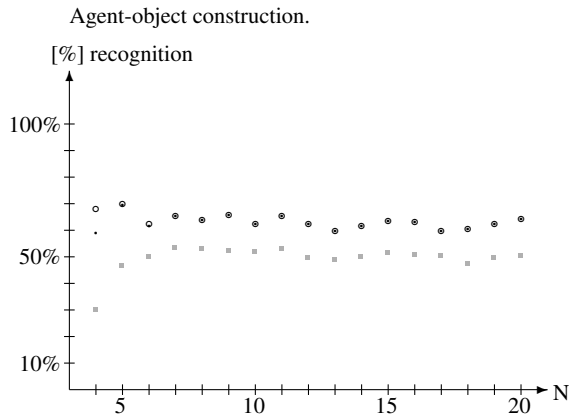
QUESTION: (3b) $\sharp see_{obj}$

ANSWER: (1b)

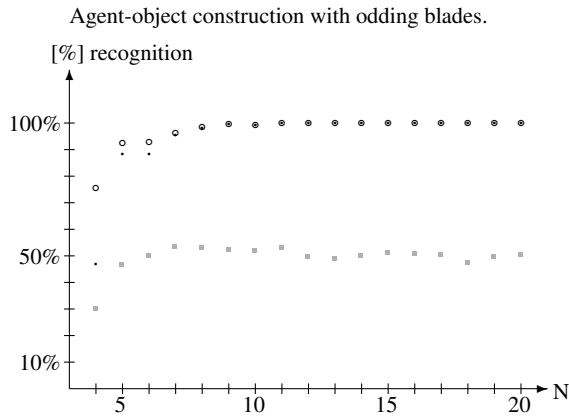
NUMBER OF TRIALS: 500

- Inner product
- Euclidean measure
- Hamming measure

(right-hand-side questions)



N	Hamming	Euclidean	Inner pr.
4	59.0%	68.0%	29.8%
5	69.6%	70.0%	46.4%
6	61.6%	62.2%	49.8%
7	65.4%	65.4%	53.4%
8	63.8%	63.8%	52.8%
9	65.8%	65.8%	52.2%
10	62.2%	62.2%	51.8%
11	65.4%	65.4%	53.0%
12	62.2%	62.2%	49.6%
13	59.8%	59.8%	48.8%
14	61.6%	61.6%	49.8%
15	63.6%	63.6%	51.2%
16	63.0%	63.0%	50.6%
17	59.6%	59.6%	50.4%
18	60.6%	60.6%	47.4%
19	62.2%	62.2%	49.6%
20	64.4%	64.4%	50.4%



N	Hamming	Euclidean	Inner pr.
4	46.8%	75.4%	26.8%
5	88.2%	92.4%	40.4%
6	88.4%	93.0%	45.6%
7	95.6%	96.2%	66.4%
8	98.2%	98.4%	78.0%
9	99.8%	99.8%	83.8%
10	99.4%	99.2%	90.6%
11	100.0%	100.0%	93.6%
12	100.0%	100.0%	91.6%
13	100.0%	100.0%	93.6%
14	100.0%	100.0%	94.2%
15	100.0%	100.0%	95.0%
16	100.0%	100.0%	92.6%
17	100.0%	100.0%	95.0%
18	100.0%	100.0%	93.6%
19	100.0%	100.0%	92.8%
20	100.0%	100.0%	93.4%

Fig. 11 Recognition test via inner product, Hamming and Euclidean measure for (3b) $\sharp see_{obj}$.

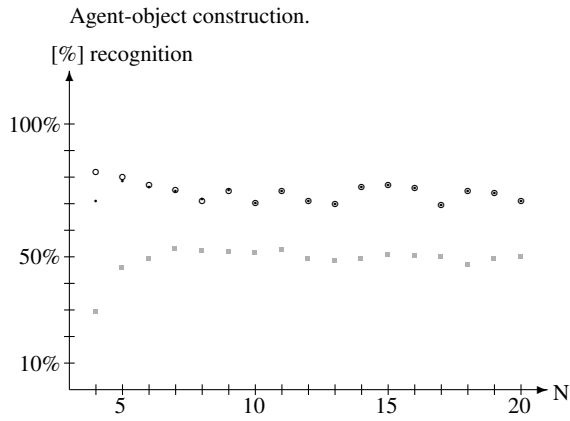
QUESTION: (5b) $\sharp see_{obj}$

ANSWER: (4b)

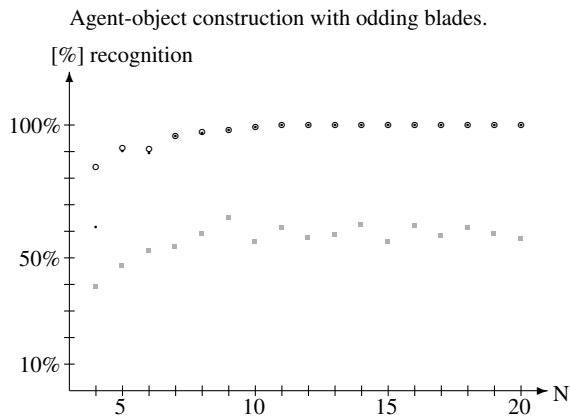
NUMBER OF TRIALS: 500

- Inner product
- Euclidean measure
- Hamming measure

(right-hand-side questions)



N	Hamming	Euclidean	Inner pr.
4	71.0%	81.8%	40.0%
5	78.6%	80.2%	47.0%
6	76.4%	77.0%	55.8%
7	74.6%	75.2%	57.0%
8	71.6%	71.2%	56.0%
9	75.0%	74.8%	59.2%
10	70.2%	70.2%	54.6%
11	74.8%	74.8%	60.8%
12	71.0%	71.0%	57.0%
13	70.0%	70.0%	54.6%
14	76.2%	76.2%	61.8%
15	77.0%	77.0%	60.0%
16	76.0%	76.0%	62.6%
17	69.6%	69.6%	52.2%
18	74.8%	74.8%	59.8%
19	74.0%	74.0%	55.2%
20	71.2%	71.2%	57.0%



N	Hamming	Euclidean	Inner pr.
4	61.6%	84.2%	39.0%
5	90.4%	91.2%	46.8%
6	89.6%	90.8%	52.6%
7	96.0%	95.8%	54.2%
8	97.0%	97.4%	58.8%
9	98.2%	98.2%	64.8%
10	99.2%	99.2%	55.8%
11	100.0%	100.0%	61.2%
12	100.0%	100.0%	57.6%
13	100.0%	100.0%	58.4%
14	100.0%	100.0%	62.2%
15	100.0%	100.0%	55.8%
16	100.0%	100.0%	61.8%
17	100.0%	100.0%	58.2%
18	100.0%	100.0%	61.0%
19	100.0%	100.0%	58.8%
20	100.0%	100.0%	57.2%

Fig. 12 Recognition test via inner product, Hamming and Euclidean measure for (5b) $\sharp see_{obj}$.

ing the distance between them, therefore one should avoid calling those measures “metrics”.

4.4 Performance of Hamming and Euclidean measures

In this Section we present some test results comparing the effectiveness of Hamming and Euclidean measures against the computation of similarity by inner product. These tests are conducted on the data set presented in Table 1. Once the inner product test indicates more than one potential answer, Hamming and Euclidean measures are employed upon the subset of the potential answers — not upon the whole clean-up memory. Figures 10 through 12 show test results for sentences with various numbers of blades using two types of construction: agent-object construction and agent-object construction with odding blades.

There was no significant difference between results obtained using the agent-object construction with odding blades and those obtained with the help of Plate construction, therefore results for Plate construction are not presented in the diagrams. Nevertheless, it is more in the spirit of distributed representations to use agent-object construction with odding blades since the additional blade is drawn at random, whereas the use of Plate construction makes data more predictable. Poor recognition in case of the agent-object construction without odding blades results from complete or partial similarity cancellation.

It becomes apparent that the best types of construction of sentences for GA are agent-object construction with odding blades and the Plate construction, as they ensure that sentences have an odd number of blades. Further, it is advisable to compute similarity by the means of Hamming measure or the Euclidean measure instead of the inner product. The Euclidean measure recognizes 100% of items much faster (i.e. for smaller data size), but for large data size both measures behave identically. Therefore Hamming measure should be used to calculate similarity since its computation requires less time. The success of those measures is due to the fact that the differences between matrices or their signatures lessen the similarity, whereas differences in blades did not lessen the value of the inner product considerably.

5 The Average Number of Potential Answers

Our point of interest in this Section will be to analyze the influence of accidental blade equality on the number of potential answers under agent-object construction with appropriate-hand-side reversed questions. The following estimates assume *ideal conditions*, i.e. no two chunks of a sentence are identical up to a constant at any time. Intuitively, such conditions could be met for sufficiently large lengths of the input vectors, whereas vectors of short length will with high probability be linearly dependent. We will estimate the average number of times that a nonzero inner

product comes up when a noisy output is compared with items stored in the clean-up memory. We will deal with the following issues:

- How often does the system produce identical blades representing atomic objects?
- How often does the system produce identical sentence chunks from different blades?
- How do the two above problems affect the number of potential answers?

Let V be the the set of multivectors over \mathbb{R}^N stored in the clean-up memory and let $\omega(V)$ be the maximum number of blades stored in a multivector in V . The set of all multivectors having the number of blades equal to k is denoted by S_k (S_1 being the set of atomic objects). Naturally, $V = S_1 \cup \dots \cup S_{\omega(V)}$. Let \tilde{n} be a noisy answer to some question. Under ideal conditions for every $c \in V$

$$|\langle \tilde{n} | c \rangle| \neq 0 \Leftrightarrow \tilde{n} \text{ and } c \text{ share a common blade.} \quad (70)$$

We will begin with a simple example of a multivector with one meaningful blade and L noisy blades. Let r_0, \dots, r_L be roles and f_0, \dots, f_L be fillers for some $L > 0$. Consider a question

$$(r_0 * f_0 + r_1 * f_1 + \dots + r_L * f_L) \sharp r_0 \quad (71)$$

which results in the following noisy answer

$$f_0 + \tilde{n}_1 + \dots + \tilde{n}_L, \quad \tilde{n}_i = r_0^+ * r_i * f_i, \quad 0 < i \leq L. \quad (72)$$

Surely, the original answer f_0 belongs to S_1 . Let $s \in V$ be an arbitrary element of the clean-up memory.

Case 1. Let $s \in S_1$ and $s \neq f_0$, in a sense that s might have the same blade as f_0 but is remembered under a different meaning in the clean-up memory. Using basic probability methods we obtain

$$|\langle s | (f_0 + \tilde{n}_1 + \dots + \tilde{n}_L) \rangle| \neq 0 \Leftrightarrow s = f_0 \text{ or } s = \tilde{n}_1 \text{ or } \dots \text{ or } s = \tilde{n}_L, \quad (73)$$

$$P[s = f_0 \text{ or } s = \tilde{n}_1 \text{ or } \dots \text{ or } s = \tilde{n}_L] = \frac{L+1}{2^N}. \quad (74)$$

Since all blades in S_1 are chosen independently, the following is true

$$\sum_{s \in S_1, s \neq f_0} P[s = f_0 \text{ or } s = \tilde{n}_1 \text{ or } \dots \text{ or } s = \tilde{n}_L] = \frac{(|S_1| - 1)(L+1)}{2^N}. \quad (75)$$

Case 2. Let $s \in S_k$ for some $1 < k \leq \omega(V)$ be a multivector made of k blades, $s = s_1 + \dots + s_k$. Since

$$P[s \text{ does not contain any of } \{f_0, \tilde{n}_1, \dots, \tilde{n}_L\}] = (1 - \frac{L+1}{2^N})^k, \quad (76)$$

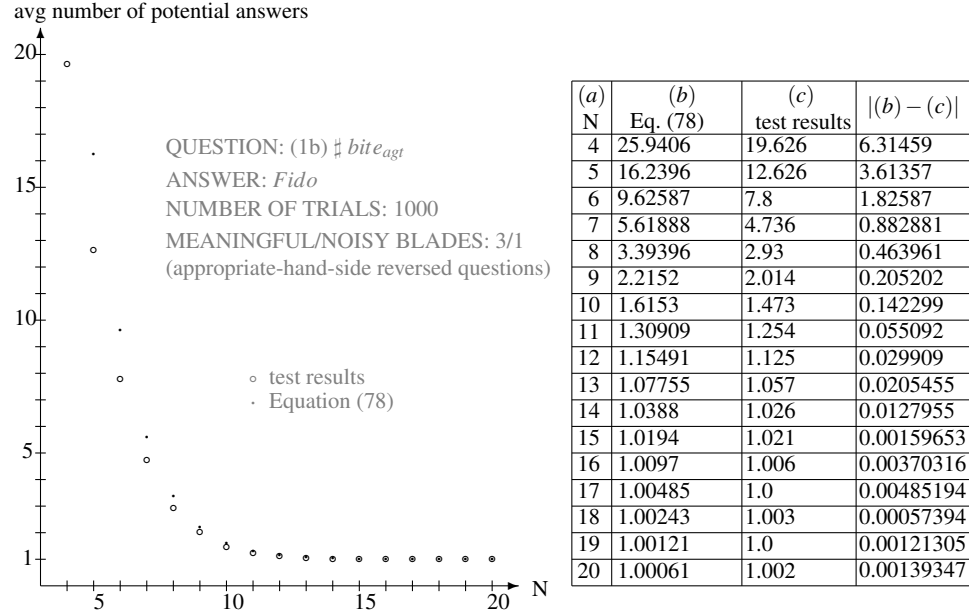


Fig. 13 Average number of potential answers per 1000 trials with a 1:3 meaningful-to-noisy blades ratio.

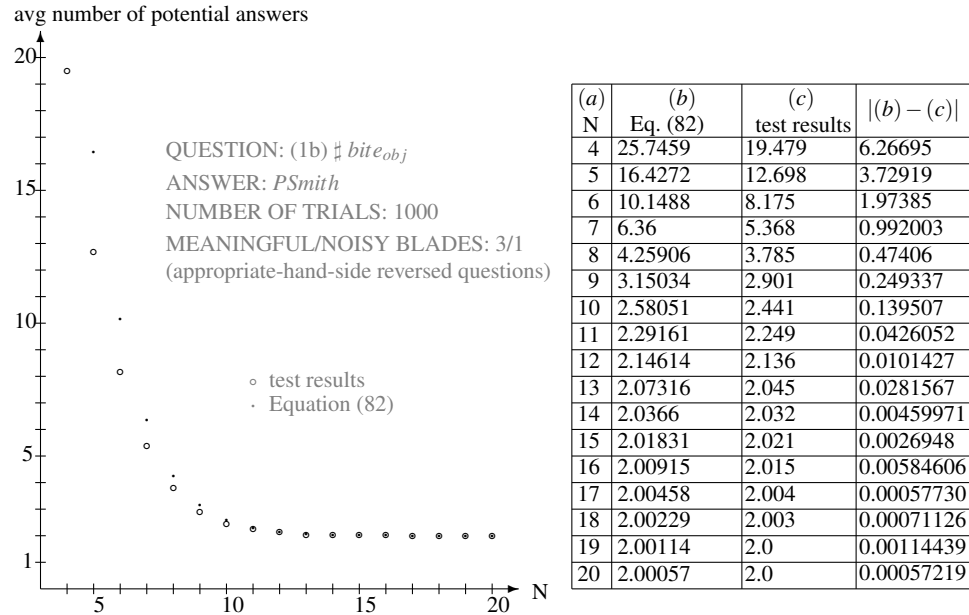


Fig. 14 Average number of potential answers per 1000 trials with a 3:1 meaningful-to-noisy blades ratio.

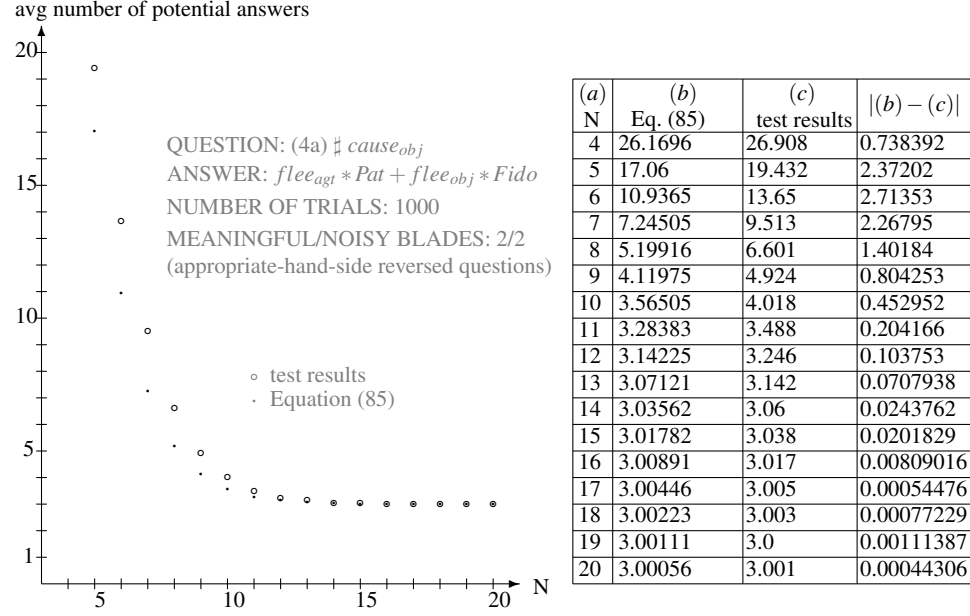


Fig. 15 Average number of potential answers per 1000 trials with a 2:2 meaningful-to-noisy blades ratio.

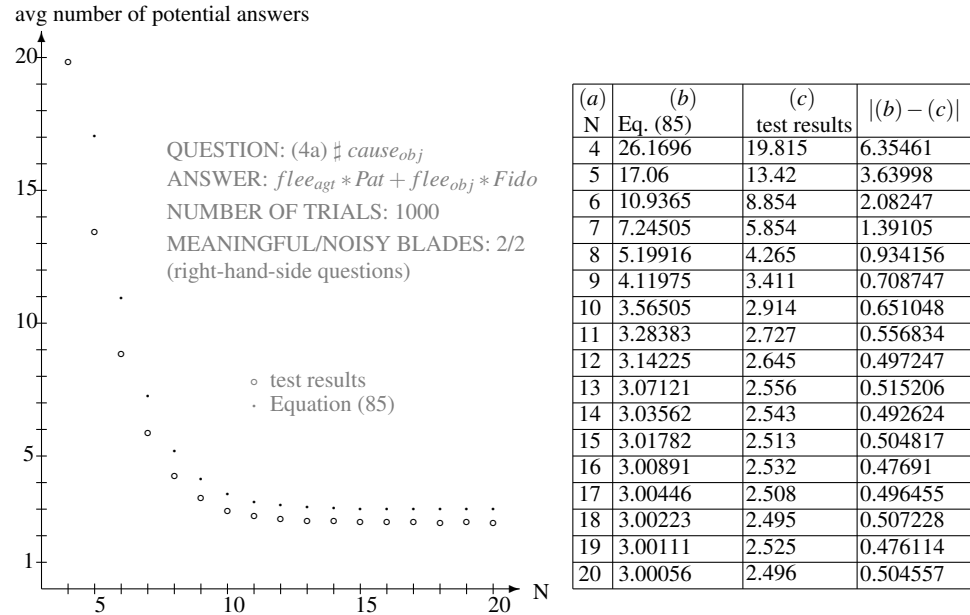


Fig. 16 Average number of potential answers per 1000 trials with a 2:2 meaningful-to-noisy blades ratio (right-hand-side questions).

we receive the following formula

$$\sum_{s \in S_k} \mathbb{P}[s \text{ contains at least one of } \{f_0, \tilde{n}_1, \dots, \tilde{n}_l\}] = |S_k| \left(1 - \left(1 - \frac{L+1}{2^N}\right)^k\right). \quad (77)$$

Thus, when probing for an answer of $f_0 + \tilde{n}_1 + \dots + \tilde{n}_L$, $L > 0$, we are likely to receive an average of

$$1 + \frac{(|S_1| - 1)(L + 1)}{2^N} + \sum_{k=2}^{\omega(V)} |S_k| \left(1 - \left(1 - \frac{L+1}{2^N}\right)^k\right) \quad (78)$$

potential answers.

Figure 13 shows test results compared with exact values given by Equation (78) for noisy answers containing one meaningful blade and three noisy blades. Note that Equation (78) is also valid for right-hand-side questions.

The situation becomes more complex when we are to deal with answers having more than one blade. Although items in S_1 are always chosen independently, we cannot say the same about items belonging to S_i , $1 < i \leq \omega(V)$, since the sentence set is chosen by the experimenter. Let us consider the following question

$$(bite_{agt} * Fido + bite_{obj} * PSmith) \# bite_{obj} \quad (79)$$

yielding an answer of four blades

$$name * Pat + sex * male + age * 66 + bite_{obj}^+ * bite_{agt} * Fido. \quad (80)$$

Clearly, the correct answer ($PSmith$) belongs to S_3 , but there is one other element of the clean-up memory listed in Table 1 that contains a portion of $PSmith$'s blades — *DogFido*

$$\begin{aligned} & class * animal + type * dog + taste * chickenlike \\ & + name * Fido + age * 7 + sex * male + occupation * pet, \end{aligned} \quad (81)$$

the common blade being $sex * male$. We have two answers that under ideal conditions will surely result in a nonzero inner product: the correct answer in S_3 and a potential answer in S_7 . By calculations analogous to those leading to Equation (78), the average number of answers giving a nonzero inner product for the above example is

$$2 + \frac{4|S_1|}{2^N} + \sum_{k \in \{3,7\}} (|S_k| - 1) \left(1 - \left(1 - \frac{4}{2^N}\right)^k\right) + \sum_{k=2, k \notin \{3,7\}}^{\omega(V)} |S_k| \left(1 - \left(1 - \frac{4}{2^N}\right)^k\right). \quad (82)$$

The number of meaningful blades in this example is odd, therefore Equation (82) is also valid for right-hand-side questions. Figure 14 shows test results compared with exact values computed by Equation (82).

Let us consider another example — now the question is

$$(4a) \# \text{cause}_{obj} \quad (83)$$

and the answer has a 2:2 meaningful-to-noisy blades ratio

$$\text{flee}_{agt} * \text{Pat} + \text{flee}_{obj} * \text{Fido} + \text{cause}_{obj}^+ * (\text{bite}_{agt} * \text{Fido} + \text{bite}_{obj} * \text{Pat}). \quad (84)$$

Apart from the correct answer in S_2 ($\text{flee}_{agt} * \text{Pat} + \text{flee}_{obj} * \text{Fido}$) there are also two potential answers belonging to the clean-up memory listed in Table 1

- sentence (2b) in S_8 — the common blade is $\text{flee}_{agt} * \text{Pat}$,
- sentence (2c) in S_4 — the common blade is $\text{flee}_{obj} * \text{Fido}$.

Therefore, the equation for calculating the estimated number of potential answers for this example takes the following form

$$3 + \frac{4|S_1|}{2^N} + \sum_{k \in \{2,4,8\}} (|S_k| - 1) \left(1 - \left(1 - \frac{4}{2^N}\right)^k\right) + \sum_{k=3, k \notin \{4,8\}}^{\omega(V)} |S_k| \left(1 - \left(1 - \frac{4}{2^N}\right)^k\right), \quad (85)$$

which is illustrated by Figure 15.

In this example test results for right-hand-side questions (see Figure 16) will differ from those obtained by formula (85) by about 0.5. That is because the scalar product of $(4a) \# \text{cause}_{obj}$ and the correct answer will produce two **1**s which, with probability 0.5, will have opposite signs and will cancel each other out. Potential answers (2b) and (2c) do not cause such problems, since the number of their blades is odd. In half the cases the number of potential answers will be 2 (sentences (2b) and (2c)) and in half the cases it will be 3 (sentences (2a), (2b) and (2c)) - achieving the average of 2.5 potential answers.

We are now ready to work out a more general formula describing the average number of potential answers for noisy statements with multiple meaningful blades. Let S and Q denote the sentence and the question respectively. Let p_k be the number of potential answers to $S \# Q$ in the subset S_k of the clean-up memory V , denote by L the number of blades in $S \# Q$ and let $p = p_1 + \dots + p_{\omega(V)}$. The formula for calculating the estimated number of potential answers to $S \# Q$ then reads

$$p + \frac{(|S_1| - p_1)L}{2^N} + \sum_{k=2}^{\omega(V)} (|S_k| - p_k) \left(1 - \left(1 - \frac{L}{2^N}\right)^k\right), \quad (86)$$

provided, that we use appropriate-hand-side reversed questions. As far as right-hand-side questions are concerned, this equation may be regarded only as the upper bound due to cancellation — for a closer estimate, one should investigate elements of the clean-up memory that have an even number of blades.

6 Comparison with Previously Developed Models

The most important performance measure of any new distributed representation model is the comparison of its efficiency in relation to previously developed models. This Section comments on test results performed on GA, BSC and HRR.

Naturally, the question of data size arises as a GA clean-up memory item may store information in more than one vector (blade), unlike in architectures known so far. Further, the preferred way of recognition for GA requires the usage of matrix signatures comprising up to $2^{1+2\lceil \frac{N}{2} \rceil}$ entries. However, since one only needs blades to calculate the matrix signatures, it has been assumed that tests comparing efficiency of various models should be conducted using the following sizes of data

- N bits for a single blade in GA,
- KN bits for a single vector in BSC and HRR,

where K is the maximum number of blades stored in a complex sentence belonging to GA's clean-up memory under agent-object construction with odding blades. For the data set presented in Table 1 the maximum number of blades is stored in items (3d) and (5b) and is equal to 13. Such an approach to the test data size will certainly prove redundant for GA sentences having a lesser number of blades, nevertheless, it is only fair to provide relatively the same data size for all compared models.

Figures 17 through 19 show comparison of performance for GA, BSC and HRR, tested sentences range in meaningful-to-noisy blades ratio from 1:2 to 7:2. Clearly, GA with the use of Hamming and Euclidean measure ensures quite a remarkable recognition percentage for sentences of great complexity and therefore — great noise, whereas the HRR model works better for statements of low complexity. There is no significant difference in performance of the BSC model as far as complexity of tested sentences is concerned. BSC does remain the best model, provided that vector lengths for BSC are sufficiently longer than that of GA. Under uniform length of vectors and blades GA recognizes sentences better than HRR or BSC, regardless of their complexity.

7 Conclusion

We have presented a new model of distributed representation that is based on the way humans think, while models developed so far were designed to use arrays of numbers mainly in order to be easily simulated by computers.

After a brief recollection of the main ideas behind the GA model, we investigated three types of sentence constructions, namely the Plate construction, the agent-object construction and the agent-object construction with odding blades. Two methods of asking questions were also investigated. As a result, in face of shortcomings of recognition based solely on the inner product, matrix representation has been employed as a recognition tool for the GA model. Using test results computed on

QUESTION: *PSmith* \sharp *name*

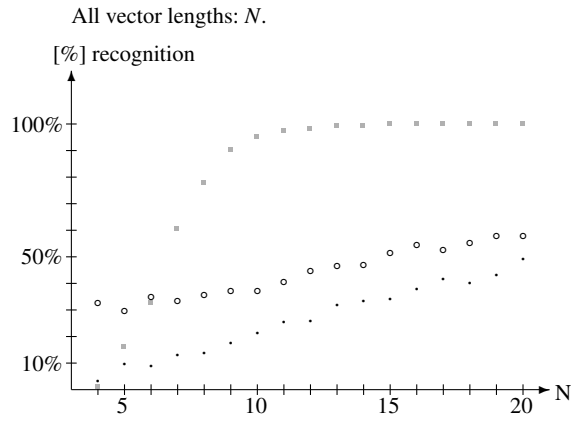
ANSWER: *Pat*

NUMBER OF TRIALS: 1000

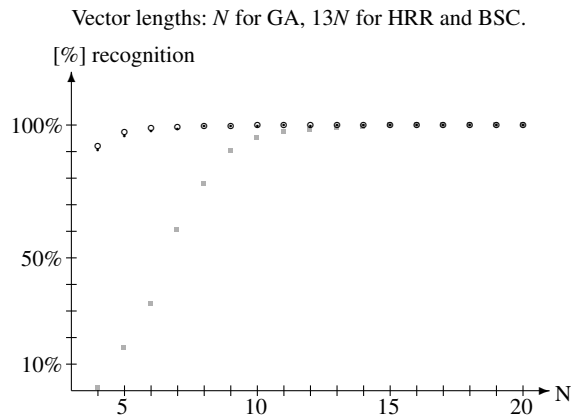
GA CONSTRUCTION: Agent-object with odding blades, right-hand-side questions

MEANINGFUL/NOISY BLADES: 1/2

- GA, Hamming measure
- BSC
- HRR



N	Hamming	HRR	BSC
4	0.8%	3.2%	32.5%
5	16.0%	9.7%	29.5%
6	32.4%	8.9%	34.9%
7	60.4%	12.9%	33.5%
8	77.6%	13.8%	35.8%
9	90.2%	17.7%	37.3%
10	95.2%	21.4%	37.3%
11	97.4%	25.5%	40.6%
12	98.2%	25.7%	44.7%
13	99.0%	32.0%	46.7%
14	99.2%	33.2%	47.1%
15	100.0%	34.3%	51.4%
16	100.0%	37.9%	54.3%
17	100.0%	41.7%	52.7%
18	100.0%	40.1%	55.3%
19	100.0%	43.3%	57.8%
20	100.0%	49.0%	57.9%



N	Hamming	HRR	BSC
4	0.8%	90.6%	92.2%
5	16.0%	95.9%	97.4%
6	32.4%	97.7%	98.7%
7	60.4%	98.6%	99.1%
8	77.6%	99.5%	99.7%
9	90.2%	99.6%	99.7%
10	95.2%	99.6%	99.9%
11	97.4%	99.9%	100.0%
12	98.2%	99.8%	100.0%
13	99.0%	100.0%	100.0%
14	99.2%	100.0%	100.0%
15	100.0%	100.0%	100.0%
16	100.0%	99.9%	100.0%
17	100.0%	100.0%	100.0%
18	100.0%	100.0%	100.0%
19	100.0%	100.0%	100.0%
20	100.0%	100.0%	100.0%

Fig. 17 Comparison of recognition for GA, BSC and HRR – *PSmith* \sharp *name*.

QUESTION: (4a) \sharp *cause_{obj}*

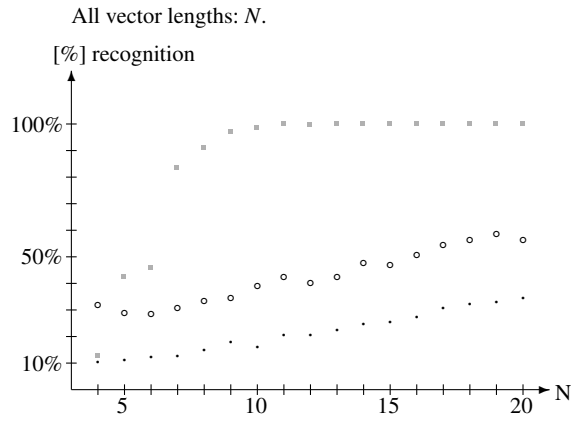
ANSWER: ANSWER: (2a)

NUMBER OF TRIALS: 1000

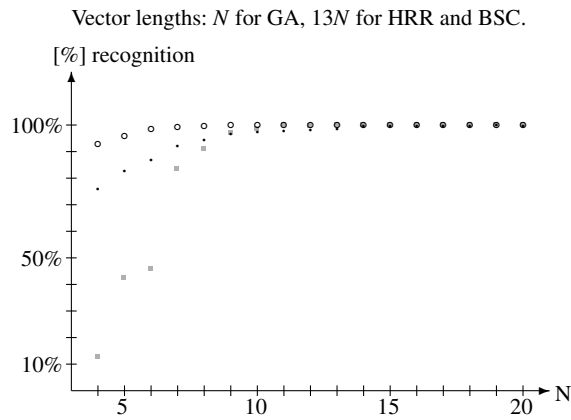
GA CONSTRUCTION: Agent-object with odding blades, right-hand-side questions

MEANINGFUL/NOISY BLADES: 3/4

- GA, Hamming measure
- BSC
- HRR



N	Hamming	HRR	BSC
4	12.6%	10.4%	32.0%
5	42.2%	11.1%	28.8%
6	45.8%	12.3%	28.4%
7	83.4%	12.7%	30.7%
8	91.0%	14.9%	33.5%
9	97.0%	18.0%	34.6%
10	98.6%	15.9%	39.0%
11	99.8%	20.7%	42.3%
12	99.6%	20.5%	40.2%
13	100.0%	22.4%	42.3%
14	99.8%	24.6%	47.7%
15	100.0%	25.3%	46.9%
16	100.0%	27.3%	50.6%
17	100.0%	30.9%	54.3%
18	100.0%	32.2%	56.2%
19	100.0%	32.9%	58.7%
20	100.0%	34.6%	56.5%



N	Hamming	HRR	BSC
4	12.6%	75.9%	92.7%
5	42.2%	82.6%	95.9%
6	45.8%	86.7%	98.6%
7	83.4%	92.1%	99.1%
8	91.0%	94.5%	99.5%
9	97.0%	96.6%	99.9%
10	98.6%	97.2%	100.0%
11	99.8%	97.9%	100.0%
12	99.6%	98.3%	100.0%
13	100.0%	98.4%	100.0%
14	99.8%	99.5%	100.0%
15	100.0%	99.5%	100.0%
16	100.0%	99.5%	100.0%
17	100.0%	99.7%	100.0%
18	100.0%	99.8%	100.0%
19	100.0%	99.9%	100.0%
20	100.0%	99.7%	100.0%

Fig. 18 Comparison of recognition for GA, BSC and HRR – (4a) \sharp *cause_{obj}*.

QUESTION: (5a) $\sharp see_{obj}$

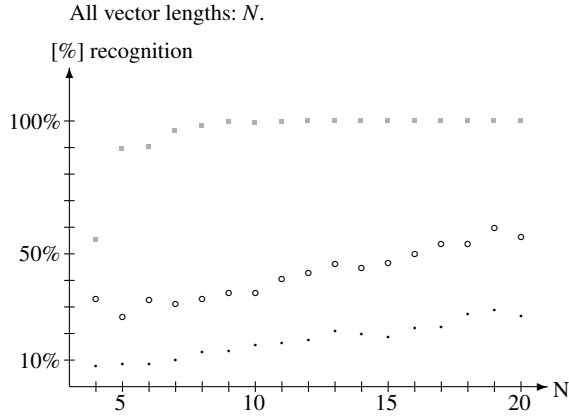
ANSWER: (4a)

NUMBER OF TRIALS: 1000

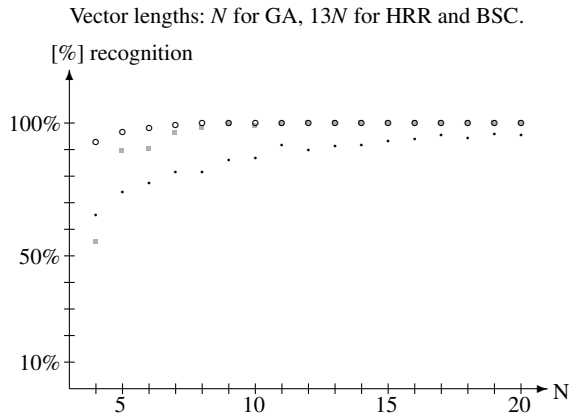
GA CONSTRUCTION: Agent-object with odding blades, right-hand-side questions

MEANINGFUL/NOISY BLADES: 7/2

- GA, Hamming measure
- BSC
- HRR



N	Hamming	HRR	BSC
4	55.2%	7.6%	32.9%
5	89.4%	8.4%	26.1%
6	90.0%	8.6%	32.5%
7	96.2%	10.2%	31.0%
8	98.0%	13.0%	32.9%
9	99.4%	13.4%	35.1%
10	99.0%	15.6%	35.3%
11	99.6%	16.6%	40.4%
12	100.0%	17.5%	42.6%
13	100.0%	21.1%	46.0%
14	100.0%	19.7%	44.8%
15	100.0%	18.8%	46.5%
16	100.0%	22.2%	49.9%
17	100.0%	22.3%	53.8%
18	100.0%	27.3%	53.6%
19	100.0%	29.0%	59.7%
20	100.0%	26.6%	56.4%



N	Hamming	HRR	BSC
4	55.2%	65.3%	93.0%
5	89.4%	74.0%	96.8%
6	90.0%	77.6%	98.0%
7	96.2%	81.6%	99.3%
8	98.0%	81.5%	100.0%
9	99.4%	85.9%	99.9%
10	99.0%	86.9%	100.0%
11	99.6%	91.6%	100.0%
12	100.0%	89.7%	100.0%
13	100.0%	91.5%	100.0%
14	100.0%	91.8%	100.0%
15	100.0%	93.3%	100.0%
16	100.0%	94.1%	100.0%
17	100.0%	95.4%	100.0%
18	100.0%	94.4%	100.0%
19	100.0%	95.8%	100.0%
20	100.0%	95.5%	100.0%

Fig. 19 Comparison of recognition for GA, BSC and HRR – (5a) $\sharp see_{obj}$.

a toy model, we have shown that Hamming and Euclidean measures of similarity perform very well under the agent-object construction with odding blades.

We also studied the ways in which the number of potential answers is affected by situations in which the system draws at random identical blades denoting different atomic objects or in which identical sentence chunks are produced from different blades. A formula estimating the number of potential counterparts of a noisy piece of information has been derived. Finally, the performance of the GA model has been compared with that of BSC and HRR models using sentences of various complexity.

Acknowledgements I would like to thank Rafał Ablamowicz and Ian Bell for many inspiring conversations that took place during the AGACSE conference in Grimma in August 2008. Furthermore, my participation in that conference would not have been possible without the support from Centrum Leo Apostel (CLEA) at the Vrije Universiteit in Brussels. I also acknowledge support from the LFPPI network.

References

1. D. Aerts, M. Czachor, B. De Moor, *On geometric-algebra representation of binary spatter codes*, preprint arXiv:cs/0610075 [cs.AI] (2006).
2. D. Aerts, M. Czachor, *Cartoon computation: Quantum-like algorithms without quantum mechanics*, J. Phys. A 40, F259 (2007).
3. D. Aerts, M. Czachor, *Tensor-product vs. geometric-product coding*, Phys. Rev. A 77, 012316 (2008), arXiv:0709.1268 [quant-ph].
4. D. Aerts, M. Czachor, B. De Moor, *Geometric Analogue of Holographic Reduced Representation*, preprint arXiv:0710.2611 (2007).
5. E. Bayro-Corrochano, *Handbook of Geometric Computing*, (Springer, Berlin 2005).
6. M. Czachor, *Elementary gates for cartoon computation*, J. Phys. A 40, F753 (2007).
7. L. Dorst, D. Fontijne, S. Mann, *Geometric Algebra for Computer Science*, (Morgan-Kaufman, Elsevier 2007).
8. D. Hestenes, *Space-Time Algebra*, (Gordon and Breach, New York, 1966).
9. D. Hestenes, G. Sobczyk, *Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics*, (Reidel, Dordrecht, 1984).
10. P. Kanerva, *Binary spatter codes of ordered k-tuples*, Artificial Neural Networks ICANN Proceedings, Lecture Notes in Computer Science vol. 1112, pp. 869-873, C. von der Malsburg et al. (Eds.), (Springer, Berlin 1996).
11. P. Kanerva, *Fully distributed representation*, Proc. 1997 Real World Computing Symposium (RWC97, Tokyo), pp. 358-365 (Real World Computing Partnership, Tsukuba-City, Japan, 1997).
12. P. Lounesto, *Clifford Algebras and Spinors*, 2nd ed., Cambridge University Press, 2001.
13. T. Plate, *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*, (CSLI Publications, Stanford, 2003).
14. P. Smolensky, *Tensor product variable binding and the representation of symbolic structures in connectionist systems*, Artificial Intelligence, 46:159216 (1990).
15. P. Smolensky, C. Dolan, *Tensor product production system: a modular architecture and representation*, Connection Science, 1(1):5368 (1989).